Chapter 4

Categories and algebra

In this chapter we develop within *ADAM* the categorical framework for manipulating inductive types in the style advocated by Hagino [37], i.e. as initial objects in a category of (F, G)-algebras, for functors F and G. This differs from the style used in the branch of mathematics called *universal algebra*, where inductive types are formed as *monads*, being themselves functors with appropriate natural transformations. In the next three chapters we shall concentrate on particular induction and recursion rules as they fit in this categorical framework.

In section 4.1 we introduce basic categorical notions in our notation, in 4.2 we introduce the categorical view of algebras over some signature. In 4.3 inductive types appear as initial F, G-algebras, and in section 4.4 we take these algebras modulo equations, which are formed either on an (abstract) syntactic or semantic level. Section 4.5 looks at the relationship with well-founded relations, and 4.6 relates the initial algebra approach to the monads of universal algebra. In 4.7, we make a comparison with the framework of Algebraic Specification.

4.1 Categorical notions

Category theory provides a number of general purpose concepts and theorems. We will use some of the most basic notions, which we introduce here in the notation of *ADAM*. For a gentle introduction to some of the basic concepts, see Rydeheard [76].

4.1.1 Categories. First the big type of *categories*. There are several equivalent definitions in use. We take a category $\mathcal{C}$ to be a type, also named $\mathcal{C}$, of *objects* together with for any pair of objects X, Y a type (or set) $X \to Y$ $\underline{\mathsf{in}}$ $\mathcal{C}$ of *morphisms* (or arrows) from X to Y, called its *hom-set*, and with associative arrow composition $\bar{\circ}$ and identity arrows Id_X. This is formalized below in the notation of paragraph 2.6.3.

$\underline{\mathsf{Define}}$ $\mathcal{C}\colon \mathbf{Cat}_i\colon \mathbf{Type}_{i+1} :=: ($
$\quad \mathcal{C}\colon \mathbf{Type}_i;$
$\quad X, Y\colon \mathcal{C} \vdash (X \to Y\ [\ \underline{\mathsf{in}}\ \mathcal{C}])\colon \mathbf{Type}_i;$
$\quad \underline{\mathsf{Variables}}\ X, Y, Z, U\colon \mathcal{C};$
$\qquad f\colon X \to Y,\ g\colon Y \to Z,\ h\colon Z \to U;$

$\mathsf{Id}_X : X \to X$,
$f \mathbin{\bar\circ} g : X \to Z$;
$\mathsf{Id}_X \mathbin{\bar\circ} f = f, \quad f \mathbin{\bar\circ} \mathsf{Id}_Y = f$,
$(f \mathbin{\bar\circ} g) \mathbin{\bar\circ} h = f \mathbin{\bar\circ} (g \mathbin{\bar\circ} h)$
)

Note that any universe of types together with functions as morphisms form a category,

$$\mathbf{TYPE}_i : \mathbf{Cat}_{i+1} \;:=\; (\mathbf{Type}_i ; (\to) ; \mathsf{I}, (\bar\circ)\,) \ .$$

4.1.2 Functors. A *functor* between two categories is a mapping of both objects and arrows that preserves identities and composition:

Define $F : (\mathcal{C} : \mathbf{Cat}) \to (\mathcal{D} : \mathbf{Cat}) =: ($
$F : \mathcal{C} \to \mathcal{D}$ in $\mathbf{TYPE}$;
Variables $X, Y, Z : \mathcal{C}$; $f : X \to Y$, $g : Y \to Z$;
$F : (X \to Y \text{ in } \mathcal{C}) \to (F.X \to F.Y \text{ in } \mathcal{D})$;
$F.\mathsf{Id}_X = \mathsf{Id}_{F.X}$,
$F.(f \mathbin{\bar\circ} g) = F.f \mathbin{\bar\circ} F.g$
)

We have identity functors I, and composition of functors denoted by reverse juxtaposition, so the type of categories with functors forms itself a (big) category:

$$\begin{array}{rrcl} & \mathsf{I}_{\mathcal{C}} : \mathcal{C} \to \mathcal{C} & := & (\mathsf{I} ; \mathsf{I}) \\ F : \mathcal{C} \to \mathcal{D},\ G : \mathcal{D} \to \mathcal{E} \ \vdash & GF : \mathcal{C} \to \mathcal{E} & := & (F \mathbin{\bar\circ} G ;\ F \mathbin{\bar\circ} G) \\ & \mathbf{CAT}_i : \mathbf{Cat}_{i+1} & := & (\mathbf{Cat}_i ; (\to) ; \mathsf{I}, (F, G :: GF)) \end{array}$$

4.1.3 Natural transformations. For categories $\mathcal{C}$, $\mathcal{D}$ and functors $F, G : \mathcal{C} \to \mathcal{D}$, a *natural transformation* $\phi : F \dot\to G$ is a family of arrows $\phi_{X:\mathcal{C}} : F.X \to G.X$ in $\mathcal{D}$ such that, for any $f : X \to Y$ in $\mathcal{C}$, one has

$$\phi_X \mathbin{\bar\circ} G.f = F.f \mathbin{\bar\circ} \phi_Y : \ \ F.X \to G.Y \ .$$

In relational notation, this is $(\phi_X, \phi_Y) \in F.f \to G.f$. Note the similarity with the typing $\phi_X : F.X \to G.X$!

As natural transformations are easily composed with each other,

$$\phi : F \dot\to G,\ \psi : G \dot\to H \ \vdash \quad \phi \mathbin{\bar\circ} \psi : F \dot\to H \ := \ (X :: \phi_X \mathbin{\bar\circ} \psi_X) \ ,$$

the class of functors $F : \mathcal{C} \to \mathcal{D}$ with natural transformations as arrows forms the *functor category* $\mathcal{D}^{\mathcal{C}}$.

Natural transformations $\phi : F \dot\to G$ can be composed with functors in two ways:

$$\begin{array}{rrcl} H : \mathcal{D} \to \mathcal{E} \ \vdash & H.\phi : HF \dot\to HG & := & (X :: H.\phi_X) \\ J : \mathcal{B} \to \mathcal{C} \ \vdash & \phi_{J.} : FJ \dot\to GJ & := & (X :: \phi_{J.X}) \end{array}$$

4.1.4 Product and exponent categories. The product $\Pi(D;\mathcal{C})$ of a family of categories is a category whose objects and arrows are tuples,

$$\begin{array}{rrcl} D\colon \mathbf{Type};\ \mathcal{C}\colon \mathbf{Cat}^D \vdash & \Pi(D;\mathcal{C}) & := (& \Pi(D;\mathcal{C}); \\ & & & X \to Y := \Pi(d\colon D :: X_d \to Y_d \ \underline{\mathsf{in}}\ \mathcal{C}_d); \\ & & & \mathsf{Id}_X := (d :: \mathsf{Id}_{X_d}), \\ & & & f \mathbin{\bar\circ} g := (d :: f_d \mathbin{\bar\circ} g_d) \\ & & &) \\ D\colon \mathbf{Type};\ \mathcal{C}\colon \mathbf{Cat} \vdash & \mathcal{C}^D\colon \mathbf{Cat} & := & \Pi(d\colon D :: \mathcal{C}) \end{array}$$

4.1.5 Dualization and initiality. For any category $\mathcal{C} = (\mathcal{C}; (\to); \mathsf{Id}, (\bar\circ))$, there is a *dual* or *opposite* category where all arrows are reversed,

$$\begin{array}{rcl} \mathcal{C}^{\mathsf{op}} & := & (\mathcal{C}; (\leftarrow); \mathsf{Id}, (\circ)) \\ \underline{\mathsf{where}}\ (X \leftarrow Y) & := & (Y \to X \ \underline{\mathsf{in}}\ \mathcal{C})\ , \\ f \circ g & := & g \mathbin{\bar\circ} f\ . \end{array}$$

So $(X \to Y \ \underline{\mathsf{in}}\ \mathcal{C}^{\mathsf{op}}) = (Y \to X \ \underline{\mathsf{in}}\ \mathcal{C})$.

An *initial* object X of a category $\mathcal{C}$ is one for which, for any object $Y\colon\mathcal{C}$, there is a unique morphism from X to Y, noted $(\![X \to Y]\!)_{\mathcal{C}}$ as in [30], or rather just $(\![Y]\!)$ when $\mathcal{C}$ and X are evident.

$$\underline{\mathsf{Define}}\ X\colon \mathsf{Init}\,\mathcal{C} :=: (X\colon\mathcal{C};\ (\![Y\colon\mathcal{C}]\!)\colon !(X \to Y)\,)$$

A *final* (or *terminal*) object of $\mathcal{C}$ is an initial object of $\mathcal{C}^{\mathsf{op}}$.

The notion of initiality, and all its derived notions, can be weakened: a *weakly initial* object X of $\mathcal{C}$ is one for which, for any object $Y\colon\mathcal{C}$ there is a morphism from X to Y, not necessarily unique.

4.1.6 Product and sum objects. A category $\mathcal{C}$ is said to have (binary) *products* iff for any pair of objects, $B\colon\mathcal{C}^2$, we have an object $B_0 \times B_1\colon\mathcal{C}$ and two morphisms $\pi_i\colon B_0 \times B_1 \to B_i$, such that for any $X\colon\mathcal{C}$; $p\colon (X, X) \to B$ there is a (unique) mediating morphism $\langle p\rangle = \langle p_0, p_1\rangle\colon X \to B_0 \times B_1$ characterized by

$$f = \langle p_0, p_1\rangle \ \Leftrightarrow\ \forall i\colon 2 :: f \mathbin{\bar\circ} \pi_i = p_i\ .$$

Using the algebraic terminology of section 4.2, one can say that $\langle p\rangle$ is a homomorphism

$$\langle p\rangle\colon (X; p) \to (B_0 \times B_1;\ \pi)\ ,$$

so that $(B_0 \times B_1;\ \pi)$ is the final object in the category of algebras $(X; p)$.

More generally, given a type N, category $\mathcal{C}$ has *products over* N iff for any tuple $B\colon\mathcal{C}^N$ there is a final object $(\Pi(N; B);\ \pi)$ in the category of algebras $(X\colon\mathcal{C};\ p\colon\Delta.X \to B)$. Here, $\Delta\colon\mathcal{C} \to \mathcal{C}^N$ is the *diagonal functor* $X \mapsto (i :: X)$.

These are called Σ-*algebras*, or (F,G)-algebras when N and M are evident. As a special case, if $N = M$ and the codomain functor G is the identity, one speaks about F-*algebras*, and their type is noted $\mathbf{Alg}\,F$.

The type of *homomorphisms* between two Σ-algebras $(T;\phi)$, $(U;\psi)$ is the subtype of those arrows $f\colon T \to U$ (that is, tuples of functions $f_i\colon T_i \to U_i$ for $i\colon N$) that preserve the operations,

$$(T;\phi) \to (U;\psi) \;:=\; \{\, f\colon T \to U \;\underline{\mathsf{in}}\; \mathbf{TYPE}^N \;|\colon\; \phi \mathbin{\bar{\circ}} G.f = F.f \mathbin{\bar{\circ}} \psi \,\} \;.$$

Using the relational interpretation of '$\to$' (section 2.12.4), this condition for $f\colon T \to U$ to be a homomorphism reads

$$(\phi,\psi) \in F.f \to G.f \;.$$

Note the similarity with the type $\phi\colon F.T \to G.T$!

One has identity and composition of homomorphisms, so algebras and homomorphisms form a category $\mathbf{ALG}\,\Sigma$ for any $\Sigma\colon \mathbf{Sign}$.

This notion of algebra is easily generalized by replacing $\mathbf{TYPE}^N$ and $\mathbf{TYPE}^M$ by arbitrary categories $\mathcal{C}$, $\mathcal{D}$. Thus one has signatures $\Sigma = (\mathcal{C},\mathcal{D}\colon \mathbf{Cat};\, F,G\colon \mathcal{C} \to \mathcal{D})$ and Σ-algebras $(T\colon \mathcal{C};\; \phi\colon F.T \to G.T \;\underline{\mathsf{in}}\; \mathcal{D})$. This corresponds to the notion of F,G-dialgebra of Hagino [37].

Example 4.1 The ring of naturals with zero, one, addition and multiplication,

$$(\mathbb{N}; \mathsf{K}\,0, \mathsf{K}\,1, (+), (\cdot)) \;,$$

forms an algebra of signature

$$\Sigma \;:=\; (N := 1,\; M := 4;\; F.X := (1, 1, X^2, X^2),\; G.X := (X, X, X, X)) \;. \tag{4.2}$$

This same algebra can also be given as an F-algebra, where $F.X := 1+1+X^2+X^2$, using the fact that the type of morphisms $A \to (X,X)$ $\underline{\mathsf{in}}$ $\mathbf{TYPE}^2$ is (naturally) isomorphic to $A_0 + A_1 \to X$ $\underline{\mathsf{in}}$ $\mathbf{TYPE}$.

4.3 Initial algebras, catamorphisms

Let $\Sigma = (\mathcal{C},\mathcal{D}; F,G)$ be a (generalized) signature. According to the definition of initial objects in 4.1.5, a Σ-algebra $(T;\tau)$ is initial iff there exists a unique homomorphism (noted $(\!|U;\psi|\!)$) from $(T;\tau)$ to any other Σ-algebra $(U;\psi)$, i.e.

$$(\!|U;\psi|\!) :\;\; !(\,(T;\tau) \to (U;\psi)\,) \;.$$

Such homomorphisms, further abbreviated to $(\!|\psi|\!)$, are called *catamorphisms* as in the Bird-Meertens formalism [57]. $(\!|\psi|\!)$ satisfies the property that, for all $f\colon T \to U$,

$$\tau \mathbin{\bar{\circ}} G.f = F.f \mathbin{\bar{\circ}} \psi \;\Leftrightarrow\; f = (\!|\psi|\!) \tag{4.3}$$

from which one has immediately the following *characteristic equation* of $(\!|\psi|\!)$:

$$\tau \mathbin{\bar{\circ}} G.(\!|\psi|\!) \;=\; F.(\!|\psi|\!) \mathbin{\bar{\circ}} \psi \;.$$

Example 4.2 We can build an initial Σ-algebra where Σ is given by (4.2). Take the type A^* of strings over the alphabet $A := \{\mathtt{0}, \mathtt{1}, \mathtt{+}, \mathtt{*}\}$, and the operations

$$\begin{aligned} z.0 &:= \mathtt{0} \\ o.0 &:= \mathtt{1} \\ x \oplus y &:= \mathtt{+}xy \\ x \otimes y &:= \mathtt{*}xy \end{aligned}$$

Let $T : \subseteq A^*$ be the least subset that is closed under these operations. Then by theorem 4.3 below, the algebra $(T; z, o, (\oplus), (\otimes))$ is an initial object of the category $\mathbf{ALG}\,\Sigma$. That is, it has a unique arrow f to any other algebra $(U; \psi)$ in this category, inductively defined by equations like $f.(\mathtt{+}xy) = \psi_2.(f.x, f.y)$ for $x, y : \in T$.

Thus, initial algebras, when they exist, are often thought of as sets of syntactic terms.

Example 4.3 From the primitive recursion principle (3.3) we get immediately the following *iteration principle*.

$$\frac{\begin{array}{l} U : \mathbf{Type} \\ b : U \\ g : U \to U \end{array}}{\exists! f : \mathbb{N} \to U :: f.0 = b \wedge f.\mathsf{s}\, n = g.(f.n)} \tag{4.4}$$

Now this says exactly that the natural numbers, with zero and successor, form an initial algebra $\Upsilon := (\mathbb{N}; \mathsf{K}\,0, \lambda\,\mathsf{s})$ of signature

$$\Sigma \;:=\; (1, 2;\; F.X := (1, X),\; G.X := (X, X)) \;,$$

for the equations say that f is a homomorphism from Υ to $(U; \mathsf{K}\,b, g)$. Equivalently, $(\mathbb{N}; [\mathsf{K}\,0, \lambda\,\mathsf{s}])$ is an initial F-algebra where $F.X := 1 + X$. We will often omit the brackets in such cases, writing just $(\mathbb{N}; \mathsf{K}\,0, \lambda\,\mathsf{s})$.

We will see in chapter 6 that the iteration principle in generalized form is equivalent to other recursion principles. (End of example)

Many other inductive types also form initial F-algebras for some F. In fact, we can take the notion of initial algebra in the category $\mathbf{TYPE}^N$ as our basic notion of inductive type.

Example 4.4 The algebra of lists of example 3.2,

$$(\mathsf{Clist}\, A; [\mathsf{K}\,\Box, +\!\!<]) \;,$$

is the initial F-algebra where $F.X := 1 + A \times X$. And the algebra of rose trees and forests of example 3.5,

$$(\mathsf{RTree}\, A, \mathsf{Forest}\, A;\; (\surd), [\mathsf{K}\,\Box, (+\!\!<)]) \;,$$

is the initial F-algebra where $F : \mathbf{TYPE}^2 \to \mathbf{TYPE}^2$ is

$$F.(X, Y) := (A \times Y, 1 + (X \times Y)) \;.$$

(End of example)

Identity is a catamorphism, and constructors are always isomorphisms here:

Theorem 4.1 (Lambek's lemma) *For any initial F-algebra $(T;\tau)$, Id is the catamorphism in $(T;\tau) \to (T;\tau)$.*

Proof. $\tau \mathbin{\bar\circ} \mathsf{Id} = F.\mathsf{Id} \mathbin{\bar\circ} \tau$, so Id is a homomorphism and hence the unique one. ■

The main use of this fact is in proving that a morphism $f: T \to T$ equals identity: $f = \mathsf{Id} \Leftrightarrow \tau \mathbin{\bar\circ} f = F.f \mathbin{\bar\circ} \tau$.

Theorem 4.2 *The constructor $\tau: F.T \to T$ of an initial F-algebra (in any category) is an isomorphism.*

Proof. We seek to define some $\delta: T \to F.T$. It should be a pre-inverse of τ:

$$\begin{array}{lll} & \delta \mathbin{\bar\circ} \tau = \mathsf{Id} & \\ \Leftrightarrow & \delta \mathbin{\bar\circ} \tau \in (T;\tau) \to (T;\tau) & \{\text{theorem 4.1}\} \\ \Leftrightarrow & \tau \mathbin{\bar\circ} (\delta \mathbin{\bar\circ} \tau) = F.(\delta \mathbin{\bar\circ} \tau) \mathbin{\bar\circ} \tau & \\ \Leftarrow & \tau \mathbin{\bar\circ} \delta = F.\delta \mathbin{\bar\circ} F.\tau & \\ \Leftrightarrow & \delta \in (T;\tau) \to (F.T; F.\tau) & \end{array}$$

So taking for δ the catamorphism $(\!|F.T; F.\tau|\!)$ will do. It remains to check that this δ is a post-inverse as well:

$$\begin{array}{lll} & \tau \mathbin{\bar\circ} \delta = \mathsf{Id} & \\ \Leftrightarrow & F.\delta \mathbin{\bar\circ} F.\tau = \mathsf{Id} & \{\tau \mathbin{\bar\circ} \delta = F.\delta \mathbin{\bar\circ} F.\tau \text{ above}\} \\ \Leftrightarrow & F.(\delta \mathbin{\bar\circ} \tau) = F.\mathsf{Id} & \{\text{functors}\} \\ \Leftarrow & \delta \mathbin{\bar\circ} \tau = \mathsf{Id} & \\ \Leftrightarrow & \mathsf{True} & \{\delta \text{ is a pre-inverse}\} \end{array}$$

Thus, we can indeed use $\tau^\cup := (\!|F.T; F.\tau|\!)$. ■

We now prove that an F-algebra being initial coincides with having no junk and no confusion. Let F be a functor in **TYPE**, extended to subsets (paragraph 4.1.7) and $(T;\tau)$ an F-algebra; we say that $(T;\tau)$ has *no confusion* iff τ is injective, i.e. $\tau \cdot \tau^\cup = \mathsf{Id}_{F.T}$, and *no junk* iff T is minimal, i.e. iff for all $S: \subseteq T$:

$$\tau \in F.S \to S \;\Rightarrow\; T \subseteq S \;. \tag{4.5}$$

Theorem 4.3 *F-algebra $(T;\tau)$ is initial iff it has no junk and no confusion.*

Proof. $\Rightarrow$: Let $(T;\tau)$ be initial, then theorem 4.2 comprises that τ is injective. Furthermore, assume $\tau \in F.S \to S$, then $(\!|\tau|\!) \in (T;\tau) \to (S;\tau)$. But theorem 4.1 says $(\!|\tau|\!) = \mathsf{I}$, so $T \subseteq S$.

$\Leftarrow$: Let $(U;\psi)$ be another F-algebra; we seek to define the unique homomorphism $f: (T;\tau) \to (U;\psi)$. The homomorphism condition $(\tau,\psi) \in F.f \to f$ gives rise to the inductive definition of f as being the smallest subset $f: \subseteq T \times U$ such that:

$$\forall g: \subseteq T \times U;\; g \text{ single-valued};\; (x,y): \in F.g :: \; g \subseteq f \Rightarrow (\tau.x, \psi.y) \in f \;.$$

It then follows from minimality of T and injectivity that f is total and single-valued, using (4.5) with

$$S := \{\, x{:}\,T \mid{:}\, \exists! f[x] \} \ .$$

One clearly has that f is a homomorphism $f{:}\,(T;\tau) \to (U;\psi)$, and one proves by extensionality and minimality that f equals any other such homomorphism.

■

4.4 Algebras with equations

Some types that have an inductive character, can be seen as F-algebras, but not initial ones, because they violate the no-confusion condition. These can often be described by using a subcategory of algebras that satisfy certain equations.

Example 4.5 Considering example 3.4, the type of joinlists with its constructors, note that $(\mathsf{JList}\,A; [\mathsf{K}\,\Box, \diamond, (+\!\!+)])$, is an F-algebra with $F.X := 1 + A + X^2$. But, as its constructors are not injective, we see that this algebra is not initial in $\mathbf{ALG}\,F$.

One can, however, form the subcategory of those F-algebras $(T; [\mathsf{K}\,e, f, (\oplus)])$ that satisfy the equations

$$\begin{aligned} s{:}\,T &\ \vdash\ e \oplus s = s,\ s \oplus e = s \\ s,t,u{:}\,T &\ \vdash\ (s \oplus t) \oplus u = s \oplus (t \oplus u). \end{aligned}$$

One may check that the algebra of joinlists is initial in this subcategory.

(End of example)

Now, what is the general form of a family of equations? The first idea is to view an equation as a pair of "syntactic" terms with free variables. We shall introduce syntactic terms in a rather abstract style, that allows for infinitary terms.

The second, even more abstract, view of equations is that an equation may be anything that establishes in a uniform way, for any algebra of the given signature, a binary relation on the carrier of the algebra. This idea gives rise to the notion of semantic equations.

We took the distinction between syntactic and semantic equations from Manes [54]. Fokkinga [30, chapter 5] introduced "transformers" and "laws" to describe equations; these are in fact a slight generalization of Manes' semantic operations and equations.

4.4.1 Syntactic terms. Let $\Sigma = (N, M; F, G)$ be a signature, so $\mathcal{C} = \mathbf{Type}^N$; we can give an inductive definition of the sets of terms for this signature Σ. First, fix for each carrier index $i{:}\,N$ the set (type) V_i of (substitutable) variable names for this carrier. So V is an object in $\mathcal{C}$. We define the type of syntactic terms over V for carrier i, $(\mathsf{T}.V)_i$, as follows:

1. The sets of variable names are embedded in the sets of terms through $\eta_V{:}\,V \to \mathsf{T}.V$ <u>in</u> $\mathbf{TYPE}^N$

2. For each operation index $j{:}\,M$, there is a syntactic operation τ_{Vj} building composite terms, so that $\tau_V{:}\,F\mathsf{T}.V \to G\mathsf{T}.V$ <u>in</u> $\mathbf{TYPE}^M$

3. These terms are all distinct and there are no more, i.e., $(\mathsf{T}.V; \tau_V, \eta_V)$ is an initial algebra of signature

$$\Sigma' := (N, M+N; \langle F, \mathsf{K}\,V\rangle, \langle G, \mathsf{I}\rangle)$$

(for simplicity, we identify $\mathbf{Type}^{M+N}$ with $\mathbf{Type}^M \times \mathbf{Type}^N$ as in 2.12.5.)

4. The choice of $\mathsf{T}.V$ is uniform with respect to V, i.e., $\mathsf{T}\colon \mathcal{C} \to \mathcal{C}$ is a functor and τ and η are natural transformations $\tau\colon F\mathsf{T} \dot{\to} G\mathsf{T}$ and $\eta\colon \mathsf{I} \dot{\to} \mathsf{T}$.

Given any Σ-algebra $(X;\phi)$ and a valuation of the variables $v\colon V \to X$ $\underline{\mathsf{in}}$ $\mathcal{C}$, one can interpret terms over V as denoting elements of X via the catamorphism $(\!|\phi, v|\!)\colon \mathsf{T}.V \to X$. Check that $(X;\phi,v)$ is another Σ'-algebra!

4.4.2 Syntactic equations. For a syntactic *equation* over signature Σ one needs two terms for one common carrier index i and over a common set of variables $V\colon \mathbf{Type}^N$. We require that each equation has its own set of (relevant) variables V, in order that a valuation has to specify values only for relevant variables.

Thus, an equation is an instance of

$$(V\colon \mathbf{Type}^N;\ i\colon N;\ s,t\colon (\mathsf{T}.V)_i)$$

where $(\mathsf{T}.V;\tau,\eta)$ is the term algebra over V. An algebra $(X;\phi)$ *satisfies* such an equation iff, for all valuations $v\colon V \to X$, the denoted elements are equal:

$$(\!|\phi, v|\!)_i.s = (\!|\phi, v|\!)_i.t\ ;$$

or, put differently, iff the relation

$$\{v\colon V \to X :: (\!|\phi, v|\!)^2_i.(s,t)\,\}$$

is contained in the equality relation $|{=_{X_i}}|$.

In general, one needs a family of equations, $(d\colon D :: (V_d; i_d; s_d, t_d))$, to delimit the required subcategory. For the example of joinlists above, taking $[\mathsf{K}\,e, f, (\oplus)] := \tau$, the three equations would be

$$\langle\ (1; e \oplus \eta.0, \eta.0),\quad (1; \eta.0 \oplus e, \eta.0),\quad (3; (\eta.0 \oplus \eta.1) \oplus \eta.2, \eta.0 \oplus (\eta.1 \oplus \eta.2))\ \rangle\ .$$

4.4.3 Semantic equations. In the category of types, a semantic equation should provide, for any Σ-algebra Φ, a relation $E\Phi\colon \subseteq \Phi^2$. Uniformity requires at least that any homomorphism $f\colon \Phi \to \Psi$ respects the relation, i.e. $f^2 \in E\Phi \to E\Psi$.

In an arbitrary category $\mathcal{C}$ with binary products, one can represent relations $R\colon A \sim B$ by subobjects $(H;r)\colon \mathcal{P}_{\mathcal{C}}(A \times B)$. To avoid products, we will use *spans*:

$$(H\colon \mathcal{C};\ r\colon (H,H) \to (A,B)\ \underline{\mathsf{in}}\ \mathcal{C}^2)\ .$$

In **TYPE**, such a span $(H;r)$ corresponds to the relation $\{h\colon H :: (r_0.h, r_1.h)\,\}$. We have a preorder $\leq$ on spans as on subobjects:

$$(H;r) \leq (H';r') \ :=\ \exists m\colon H \to H' :: r = (m,m) \mathbin{\bar{\circ}} r'$$

The identity relation on $A : \mathcal{C}$ is represented by the span $(A; \mathsf{Id})$.

Now, we define a *semantic equation* (or *law*) E over a signature $\Sigma = (\mathcal{C}, \mathcal{D}; F, G)$ to be, for any Σ-algebra $(X; \phi)$, a span $E(X; \phi) : X \sim X$ which is uniformly defined in the sense that

$$E(X; \phi) = (H.X; r(X; \phi))$$

for a functor $H : \mathcal{C} \to \mathcal{C}$ together with two natural transformations $r_0, r_1 : HU \dot{\to} U$, where $U : \mathbf{ALG}\,\Sigma \to \mathcal{C}$ is the *forgetful functor* $(X; \phi) \mapsto X$. That is, one has

$$r : ((X; \phi) : \mathbf{ALG}\,\Sigma \triangleright H.X \to X)^2$$

satisfying the promotion law that, for all Σ-algebras Φ, Ψ,

$$\forall f : \Phi \to \Psi \text{ in } \mathbf{ALG}\,\Sigma :: r_j\Phi \mathbin{⨾} f = H.f \mathbin{⨾} r_j\Psi \ .$$

The r_j are called *semantic operations*. Algebra $(X; \phi)$ *satisfies* equation E iff $E(X; \phi)$ is contained in the identity relation, $E(X; \phi) \leq (X; \mathsf{Id})$, which is equivalent to (4.6):

$$\begin{aligned} & (H.X; r(X; \phi)) \leq (X; \mathsf{Id}) \\ \Leftrightarrow\ & \exists m : H.X \to X :: r(X; \phi) = (m, m) \mathbin{⨾} \mathsf{Id} \\ \Leftrightarrow\ & r_0(X; \phi) = r_1(X; \phi) \end{aligned} \tag{4.6}$$

One checks easily that our first uniformity requirement, that homomorphisms respect semantic equations, is satisfied indeed. The set of all algebras that satisfy a law forms a subcategory,

$$\mathbf{ALG}(\Sigma; E) \ := \ \{\, \Phi : \mathbf{ALG}\,\Sigma \mid : r_0\Phi = r_1\Phi \,\} \ .$$

Fokkinga [30, chapter 5] calls a natural transformation $r : HU \dot{\to} JU$ a *transformer of type* $(F, G) \to (H, J)$. The current notion of semantic operation (following Manes) corresponds to transformers of type $(F, G) \to (H, \mathsf{I})$. Transformers can be composed both horizontally and vertically, which gives rise to algebraic manipulations:

$$\begin{aligned} r : (F, G) \to (H, J);\ r' : (F, G) \to (J, K) \vdash \quad r \mathbin{⨾} r' \ &:= \ (\Phi :: r\Phi \mathbin{⨾} r'\Phi) : \ (F, G) \to (H, K) \\ r : (F, G) \to (H, J);\ s : (H, J) \to (K, L) \vdash \quad sr \ &:= \ (\Phi :: s(U.\Phi; r\Phi)) : \ (F, G) \to (K, L) \end{aligned}$$

We now show that semantic equations really generalize syntactic ones, and moreover, that a single semantic equation suffices.

Theorem 4.4 *Any family of syntactic equations corresponds to a single semantic equation, provided the base category has sums and exponents.*

Proof. Given a family of syntactic equations $(d : D :: (V_d; s_d, t_d))$, we have to find a law $(H; r)$ such that

$$\forall (d : D;\ v : V_d \to X :: (\!|\phi, v|\!).s_d = (\!|\phi, v|\!).t_d\,) \ \Leftrightarrow \ r_0(X; \phi) = r_1(X; \phi) \ . \tag{4.7}$$

That's fairly simple; take

$$\begin{aligned} H.X \ &:= \ \Sigma(d : D :: X^{V_d}) \ ; \\ r(X; \phi) \ &:= \ (d; v) \mapsto ((\!|\phi, \lambda v|\!).s_d, (\!|\phi, \lambda v|\!).t_d) \ , \end{aligned}$$

then the righthand side of (4.7) unfolds to the lefthand side. ∎

We shall see in section 8.3 that for certain signatures, called polynomial, initial $(\Sigma; E)$-algebras do generally exist.

4.5 Initial algebras related to well-founded relations

For bijective F-algebras, we will define a predecessor relation that is well-founded just when the algebra is initial. We cannot do this for algebras with equations, nor does an arbitrary well-founded relation correspond to some initial algebra.

Let $F\colon \mathbf{TYPE} \to \mathbf{TYPE}$ be a functor, extended to subsets, that preserves nonempty intersections,

$$\exists A;\ X_{i:A}\colon \mathcal{P}T \vdash \quad F.\bigcap(i\colon A :: X_i) = \bigcap(i\colon A :: F.X_i)\ ,$$

and $(T;\tau)$ a bijective F-algebra. We define the set of predecessors of $x\colon T$ as the least set $X\colon \subseteq T$ for which $x \in \tau[F.X]$. So we have a relation on T:

$$|\prec x| \ :=\ \bigcap(X\colon \subseteq T \mid\colon x \in \tau[F.X])$$

We must check that this set satisfies $x \in \tau[F.X]$ itself:

$$\begin{array}{lcll} & \forall x\colon T :: x \in \tau[F.|\prec x|] & \\ \Leftrightarrow & \forall y\colon F.T :: y \in F.\bigcap(X \mid\colon \tau.y \in \tau[F.X]) & \{\tau \text{ is surjective: } x = \tau.y\} \\ \Leftrightarrow & \forall y\colon F.T :: y \in F.\bigcap(X \mid\colon y \in F.X) & \{\tau \text{ is injective}\} \\ \Leftrightarrow & \forall y\colon F.T :: y \in \bigcap(X\colon \subseteq T;\ y \in F.X :: F.X) & \{F \text{ preserves } \bigcap\} \\ \Leftrightarrow & \mathsf{True} & \end{array}$$

Theorem 4.5 *For $F, T, \tau, \prec$ as above, T has no junk iff $\prec$ is well-founded.*

Proof.

$$\begin{array}{lcll} & T \text{ has no junk} & \\ \Leftrightarrow & \forall S\colon \subseteq T;\ \tau \in F.S \to S :: T \subseteq S & \\ \Leftrightarrow & \forall S\colon \subseteq T;\ \forall(x\colon T;\ x \in \tau[F.S] :: x \in S) :: T \subseteq S & \\ \Leftrightarrow & \forall S\colon \subseteq T;\ \forall(x\colon T;\ |\prec x| \subseteq S :: x \in S) :: T \subseteq S & \{\text{See below}\} \\ \Leftrightarrow & \prec \text{ is well-founded} & \end{array}$$

It remains to prove $x \in \tau[F.S] \Leftrightarrow |\prec x| \subseteq S$.

$\Rightarrow$: Immediate, by definition of $\prec$.

$\Leftarrow$: When $|\prec x| \subseteq S$, then $\tau[F.|\prec x|\,] \subseteq \tau[F.S]$. As $x \in \tau[F.|\prec x|\,]$, we are done. ■

(Classically, $\prec$ being well-founded implies τ being surjective, so that requirement could be dropped.)

If one has an algebra with equations, τ is no longer injective so this construction of $\prec$ would not make sense. Indeed, take the initial algebra $(T; a, b, f)$ in the category of algebras

$$\{\,(T\colon \mathbf{TYPE};\ a, b\colon T,\ f\colon T \to T) \mid\colon f.a = f.b\}\ .$$

Then the set $|\prec f.a|$ of immediate predecessors of $f.a$ would have to be $\{\,a\,\}$ and $\{\,b\,\}$ at the same time.

Conversely, not all well-founded relations $(\prec) : \subseteq T^2$ correspond to some initial F-algebra with equations (or without). For, take the type $T := \{0, 1\}$ with the ordering $0 \prec 1$. A corresponding algebra should have some operation $f : T \to T$ with $f.0 = 1$. But then it would also have an element $f.1$ with $1 \prec f.1$, which T has not.

4.6 An aside: monads

Universal (categorical) algebra usually talks about inductive types in the form of monads, see Manes [54]. A nice introduction is also given by Lambek and Scott in [46, page 27–34]. As a side trip in our exposition, we summarize this concept here and establish its relationship to initial F-algebras.

A *monad* on a category $\mathcal{C}$ is a triple $(T; \eta, \mu)$ consisting of a functor and two natural transformations, typed by

$$\begin{array}{rcl} T & : & \mathcal{C} \to \mathcal{C} \\ \eta & : & \mathsf{I}_{\mathcal{C}} \dot{\to} T \\ \mu & : & TT \dot{\to} T \ , \end{array}$$

that satisfy the three equations

$$T.\eta \mathbin{\bar{\circ}} \mu = \mathsf{Id}_T = \eta_{T.} \mathbin{\bar{\circ}} \mu \ , \qquad \mu_{T.} \mathbin{\bar{\circ}} \mu = T.\mu \mathbin{\bar{\circ}} \mu \ . \tag{4.8}$$

If $\mathcal{C}$ is **TYPE**, one may think of $T.V$ as a type of structured values with sub-values drawn from V. Transformation η creates a value consisting of a single subvalue, and transformation μ merges a structured value with its structured subvalues..

Example 4.6 The monad of lists in the category of types is given by $T.X := X^*$, $\eta_X.x := \langle x \rangle$, and μ being the join function that concatenates a list of lists into a single list, so $\mu.\langle a, b \rangle = a +\!\!+ b$. To get some understanding of the monad equations (4.8), we apply them to respectively the list $\langle x, y \rangle$ and the list of lists of lists $\langle \langle a, b \rangle, \langle c, d \rangle \rangle$, and get:

$$\mu.\langle \langle x \rangle, \langle y \rangle \rangle = \langle x, y \rangle = \mu.\langle \langle x, y \rangle \rangle \ , \qquad \mu.\mu.\langle \langle a, b \rangle, \langle c, d \rangle \rangle = \mu.\langle \mu.\langle a, b \rangle, \mu.\langle c, d \rangle \rangle \ .$$

The following theorem gives a link with initial F-algebras.

Theorem 4.6 *1. For any functor F, if $(T.V; [\tau_V, \eta_V])$ is a uniformly defined initial $(F + \mathsf{K}\, V)$-algebra (so that τ and η are natural transformations), then define*

$$\mu_V : TT.V \to T.V \ := \ (\!| T.V; [\tau_V, \mathsf{Id}_{T.V}] |\!)$$

to make a monad $(T; \eta, \mu)$.

2. *Conversely, any monad $(T; \eta, \mu)$ can be made into a $T + \mathsf{K}\, V$-algebra $(T.V; [\mu_V, \eta_V])$, but not necessarily into an initial one.*

Proof 1. Check that μ_V is correctly typed (its domain is an initial $F + \mathsf{K}(T.V)$-algebra, so see that $(T.V; [\tau_V, \mathsf{Id}_V])$ is an algebra of the same signature). As μ is clearly polymorphic, it is a natural transformation by the naturality theorem D.1. Then we have to check (4.8). By the catamorphism property we have the following.

$$\tau_{T.} \mathbin{\bar{\circ}} \mu = F.\mu \mathbin{\bar{\circ}} \tau \tag{4.9}$$
$$\eta_{T.} \mathbin{\bar{\circ}} \mu = \mathsf{Id}_{T.} \tag{4.10}$$

The equality $\eta_{T.} \mathbin{\bar{\circ}} \mu = \mathsf{Id}_{T.}$ goes

$$\begin{array}{ll} & \eta_{T.} \mathbin{\bar{\circ}} \mu \\ = & \sigma_1 \mathbin{\bar{\circ}} [\tau_{T.}, \eta_{T.}] \mathbin{\bar{\circ}} \mu \\ = & \sigma_1 \mathbin{\bar{\circ}} (F.\mu + \mathsf{Id}_{T.}) \mathbin{\bar{\circ}} [\tau, \mathsf{Id}_{T.}] \quad \{\mu \text{ is a catamorphism}\} \\ = & \mathsf{Id}_{T.} \mathbin{\bar{\circ}} \sigma_1 \mathbin{\bar{\circ}} [\tau, \mathsf{Id}_{T.}] \\ = & \mathsf{Id}_{T.} \end{array}$$

Then to check $T.\eta \mathbin{\bar{\circ}} \mu = \mathsf{Id}_{T.}$, we use theorem 4.1 saying that $f = \mathsf{Id}_{T.V}$ iff

$$(F.f + \mathsf{Id}_V) \mathbin{\bar{\circ}} [\tau, \eta] \;=\; [\tau, \eta] \mathbin{\bar{\circ}} f$$

which is equivalent to

$$F.f \mathbin{\bar{\circ}} \tau = \tau \mathbin{\bar{\circ}} f \;\wedge\; \eta = \eta \mathbin{\bar{\circ}} f \;.$$

Instantiating this to $f := T.\eta \mathbin{\bar{\circ}} \mu$, we calculate first

$$\begin{array}{ll} & \tau \mathbin{\bar{\circ}} T.\eta \mathbin{\bar{\circ}} \mu \\ = & FT.\eta \mathbin{\bar{\circ}} \tau_{T.} \mathbin{\bar{\circ}} \mu \quad \{\tau \text{ is natural}\} \\ = & FT.\eta \mathbin{\bar{\circ}} F.\mu \mathbin{\bar{\circ}} \tau \quad \{(4.9)\} \\ = & F.(T.\eta \mathbin{\bar{\circ}} \mu) \mathbin{\bar{\circ}} \tau \quad \{\text{functor}\} \end{array}$$

and second

$$\begin{array}{ll} & \eta \mathbin{\bar{\circ}} T.\eta \mathbin{\bar{\circ}} \mu \\ = & \eta \mathbin{\bar{\circ}} \eta_{T.} \mathbin{\bar{\circ}} \mu \quad \{\eta \text{ is natural}\} \\ = & \eta \mathbin{\bar{\circ}} \mathsf{Id}_{T.} \quad \{(4.10)\} \\ = & \eta \end{array}$$

Finally we derive $\mu_{T.V} \mathbin{\bar{\circ}} \mu_V = T.\mu_V \mathbin{\bar{\circ}} \mu_V$ by proving that both sides are equal to $(\!|T.V; [\tau_V, \mu_V]|\!)$. Note that $f = (\!|T.V; [\tau_V, \mu_V]|\!)$ iff

$$\tau_{TT.V} \mathbin{\bar{\circ}} f = F.f \mathbin{\bar{\circ}} \tau_V \;\wedge\; \eta_{TT.V} \mathbin{\bar{\circ}} f = \mu_V \;.$$

So together we have four proof obligations to check.

$$\begin{array}{ll} & \tau_{TT.} \mathbin{\bar{\circ}} \mu_{T.} \mathbin{\bar{\circ}} \mu \\ = & F.\mu_{T.} \mathbin{\bar{\circ}} \tau_{T.} \mathbin{\bar{\circ}} \mu \quad \{(4.9)\} \end{array}$$

Similarly, $\mathcal{C}$ is said to have (binary) *sums* (or coproducts) iff $\mathcal{C}^{\mathsf{op}}$ has binary products, noted $(B_0 + B_1; \sigma)$. Given $B{:}\,\mathcal{C}^2$; $X{:}\,\mathcal{C}$; $s{:}\,B \to (X, X)$, the mediating morphism is noted as

$$[s_0, s_1]{:}\,(B_0 + B_1;\ \sigma) \to (X; s)\ .$$

Note that the category of types has products and sums over all types in the category indeed, and the notations $\langle p \rangle$ and $[s]$ were already introduced in paragraphs 2.5.3 and 2.6.1 for the mediating morphisms in this category.

4.1.7 Subobjects. For an object $A{:}\,\mathcal{C}$ of any category, we can define the category of *subobjects of* A,

$$\begin{array}{rcl} \mathcal{P}_{\mathcal{C}} A & := & \{\, (H{:}\,C;\ r{:}\,H \to A)\,\} \\ (H; r) \to (H'; r') \ \underline{\mathsf{in}}\ \mathcal{P}_{\mathcal{C}} A & := & \{\, f{:}\,H \to H' \ \underline{\mathsf{in}}\ \mathcal{C} \mid{:}\ r = f \mathbin{\circ} r' \} \end{array}$$

We define a preorder $(\leq)$ on subobjects, and any functor on $\mathcal{C}$ extends to a functor on $P_{\mathcal{C}} A$ (which preserves $(\leq)$):

$$\begin{array}{rcl} (H; r) \leq (H'; r') & := & \exists((H; r) \to (H'; r') \ \underline{\mathsf{in}}\ \mathcal{P}_{\mathcal{C}} A) \\ F.(H; r) & := & (F.H; F.r) \end{array}$$

For $\mathcal{C} := \mathbf{TYPE}$, a subobject $(H; r)$ represents the subset $\{z{:}\,H :: r.z\,\}{:}\,\mathcal{P}A$, and any subset $S{:}\,\mathcal{P}A$ is represented by a subobject $(S; \mathsf{I})$. Relation $(\leq)$ represents subset inclusion $(\subseteq)$, and extended functors preserve not only inclusions $S \subseteq S'$, but also *inclusion maps*:

$$F.(\mathsf{I}{:}\,S \to S') \ =\ \mathsf{I}{:}\,F.S \to F.S'\ . \tag{4.1}$$

Note that the latter property is not automatic for functors on the subset category $(\mathcal{P}A; (\to))$ or on **SET**, for inclusion maps are not identity arrows when $S \neq S'$.

4.2 Algebras and signatures

An *algebra* is essentially a tuple of types T_i called the *carriers* or *sorts*, where i ranges over some type N called the set of *sort names*, together with a tuple of functions ϕ_j called the *operations*, where j ranges over a type M of operation names. The domain and codomain (range) of the operations is specified by a signature.

We use a more liberal notion of signature than in the tradition of Algebraic Specification, see section 4.7. The *signature* of an algebra consists of the types N and M, and two functors $F, G{:}\,\mathbf{TYPE}^N \to \mathbf{TYPE}^M$, specifying for each operation its domain and codomain, so that $\phi_j{:}\,(F.T)_j \to (G.T)_j$.

The type of signatures is thus

$$\mathbf{Sign}{:}\,\mathbf{Type}\ :=\ \{\, (N, M{:}\,\mathbf{Type};\ F, G{:}\,\mathbf{TYPE}^N \to \mathbf{TYPE}^M)\,\}$$

and the type of algebras with a given signature $\Sigma = (N, M; F, G)$ is

$$\mathbf{Alg}\,\Sigma\ :=\ \{\, (T{:}\,\mathbf{Type}^N;\ \phi{:}\,F.T \to G.T \ \underline{\mathsf{in}}\ \mathbf{TYPE}^M)\,\}\ .$$

$$
\begin{array}{lcll}
 & F.\mu_{T.} \mathbin{\bar{\circ}} F.\mu \mathbin{\bar{\circ}} \tau & \{(4.9)\} \\
= & F.(\mu_{T.} \mathbin{\bar{\circ}} \mu) \mathbin{\bar{\circ}} \tau & , \\
 & & \\
 & \eta_{TT.} \mathbin{\bar{\circ}} \mu_{T.} \mathbin{\bar{\circ}} \mu & \\
= & \mathsf{Id}_{TT.} \mathbin{\bar{\circ}} \mu & \{(4.10)\} \\
= & \mu & , \\
 & & \\
 & \tau_{TT.} \mathbin{\bar{\circ}} T.\mu \mathbin{\bar{\circ}} \mu & \\
= & FT.\mu \mathbin{\bar{\circ}} \tau_{T.} \mathbin{\bar{\circ}} \mu & \{\tau \text{ is natural}\} \\
= & FT.\mu \mathbin{\bar{\circ}} F.\mu \mathbin{\bar{\circ}} \tau & \{(4.9)\} \\
= & F.(T.\mu \mathbin{\bar{\circ}} \mu) \mathbin{\bar{\circ}} \tau & , \\
 & & \\
 & \eta_{TT.} \mathbin{\bar{\circ}} T.\mu \mathbin{\bar{\circ}} \mu & \\
= & \mu \mathbin{\bar{\circ}} \eta_{T.} \mathbin{\bar{\circ}} \mu & \{\eta \text{ is natural}\} \\
= & \mu \mathbin{\bar{\circ}} \mathsf{Id}_{T.} & \{(4.10)\} \\
= & \mu &
\end{array}
$$

2. It is obviously a $T + \mathsf{K}\,V$-algebra. It need not be initial, as shown by the following counterexample. Let $\mathcal{C} := \mathbf{TYPE}$, $T := \mathsf{K}\,1$, let η and μ be the unique transformations to $\mathsf{K}\,1$, and $V := 1$. The carrier of the initial $T + \mathsf{K}\,V$-algebra is then 2, not 1. ■

4.7 Algebraic Specification

Let us make a few remarks about the relation between the approach sketched here and the tradition of Algebraic Specification (A.S.) as reviewed by Wirsing [87].

A.S. uses a notion of signature that is more restrictive than our categorical formulation, in that the argument and result types of operations must be sorts from the algebra itself. Such an algebra is called *plain* in 5.2.2. Furthermore, the sets of sorts, operations, and arguments of each operation are often required to be finite. Thus, a signature Σ consists of finite numbers or name sets n, m for the sorts and operations, and arities $(d_j, c_j)\colon n^* \times n$ specifying the domain and codomain of the operation named by j :

$$\Sigma \;=\; (n, m\colon \mathbb{N};\; d\colon (n^*)^m, c\colon n^m)$$

The algebras of signature Σ are given by:

$$(T\colon \mathbf{Type}^n;\; \tau_{j:m}\colon \Pi(k\colon \#d_j :: T_{(djk)}) \to T_{(cj)})$$

A data type specification is given in both approaches by means of a signature with axioms. A.S. is more liberal in that the axioms may contain inequations, conditional equations, or even unrestricted axioms in first order logic. The main difference lies in the interpretation and derivational use of such a specification.

In our approach, we define the data type explicitly to be an initial algebra of the given signature, which needs to contain only the basic constructors and if necessary additional equations. The signature should be of a form that guarantees existence of

such an algebra. We use full logic, even higher order if needed, to exploit initiality in defining derived functions and in deriving theorems. The logic may be restricted for specific purposes.

In A.S., one should distinguish between the formal application and the semantic interpretation of a specification. Formal derivations may only use the specified axioms, so that derived theorems hold for all algebras of the given signature. As a consequence, the signature must contain additional operations and axioms describing their behavior. A specification of lists for example must either contain operations yielding the head, tail, and length of a list, or a recursion operator. There is no formal guarantee that the signature axioms are consistent. The logic is often quite restricted, for example purely equational, allowing only finitary operations, and not containing function types.

One can prove consistency by meta-reasoning on the semantic level, where one distinguishes initial and terminal interpretations. For example, one can point out some of the operations as being constructors, prove that the sub-signature of these constructors has an initial model, and prove that this model has indeed operations that satisfy the remaining axioms. For terminal interpretations, see section 7.5.

A.S. has the advantage of admitting several alternative semantic concepts, like partial or continuous algebras. A simple logic is of course simpler to implement, to master, and to analyze.

4.8 Concluding remarks

Now that we have the basic categories of algebras, the stage is set for discussing the various forms that inductive and recursive definitions may take. The next chapter studies forms of inductive type definitions themselves, the subsequent one recursive function definitions over inductive types. The dual forms follow in chapter 7.